Electricity Price Forecasting by Hybrid Model

Department of Data Science and Analytics
Central University of Rajasthan

Presented By:
Nitesh Sukhwani
Pradyumna Kumar Sahoo

Introduction

In Competitive electricity markets, Electricity price forecasting is very valuable for all participants. Good forecasting plays a very important role in power investment decision and transmission expansion.

Agenda

1. Why Electricity price forecasting is difficult ?
2. Why didn’t we simply employed the statistical model??
3. Why we didn’t only use machine learning??
4. Wavelet transformation
5. Extreme Machine Learning (ELM)
6. Particle Swarm Optimisation(PSO)
7. Time- Series Analysis
8. Performance Measures
9. Results
10. Conclusion
11. References

Electricity price forecasting is difficult.Why??

1. High volatility
2. High frequency
3. Nonlinearity
4. Mean-reversion
5. Non-stationarity
6. Calendar-effect
7. Price spikes

Why didn’t we simply employed the statistical model??

Because :

1.Limited ability to capture the non-linear behavior

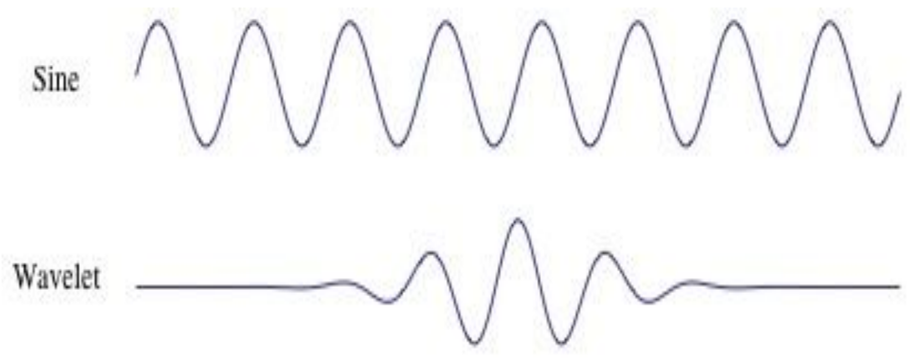
2.Rapid changes in the price signal

Why we didn't only use machine learning??

1. Selection of network structure and parameters dependent on **experience**
2. Features captured by the ANNs method may lose their value as time changes
3. **Excessive** tunable parameters
4. High possibility of entrapment in local minima
5. Over-tuning

Wavelet transformation:

- Wavelets = Mini-Waves
- Wavelet Transform tells us about the frequencies present as well as the time in which these frequencies were observed.
- It is possible to model with a finite wave which you can ‘slide’ along to time domain (Convolution)



Wavelet transformation:

- ❖ First we work on the signal with larger windows captures the larger features and then,for smaller features we use a smaller window.
- ❖ Small frequencies = high resolution in the frequency domain but less resolution in the time domain.
- ❖ On the other hand, it has large resolution in time domain for large frequencies but less in frequency domain.

The maths behind Wavelet..

$$X = \frac{1}{\sqrt{|a|}} \int_{-\infty}^{\infty} x(t)\bar{\psi}(\frac{t-\tau}{a})dt$$

The decomposed series

scale

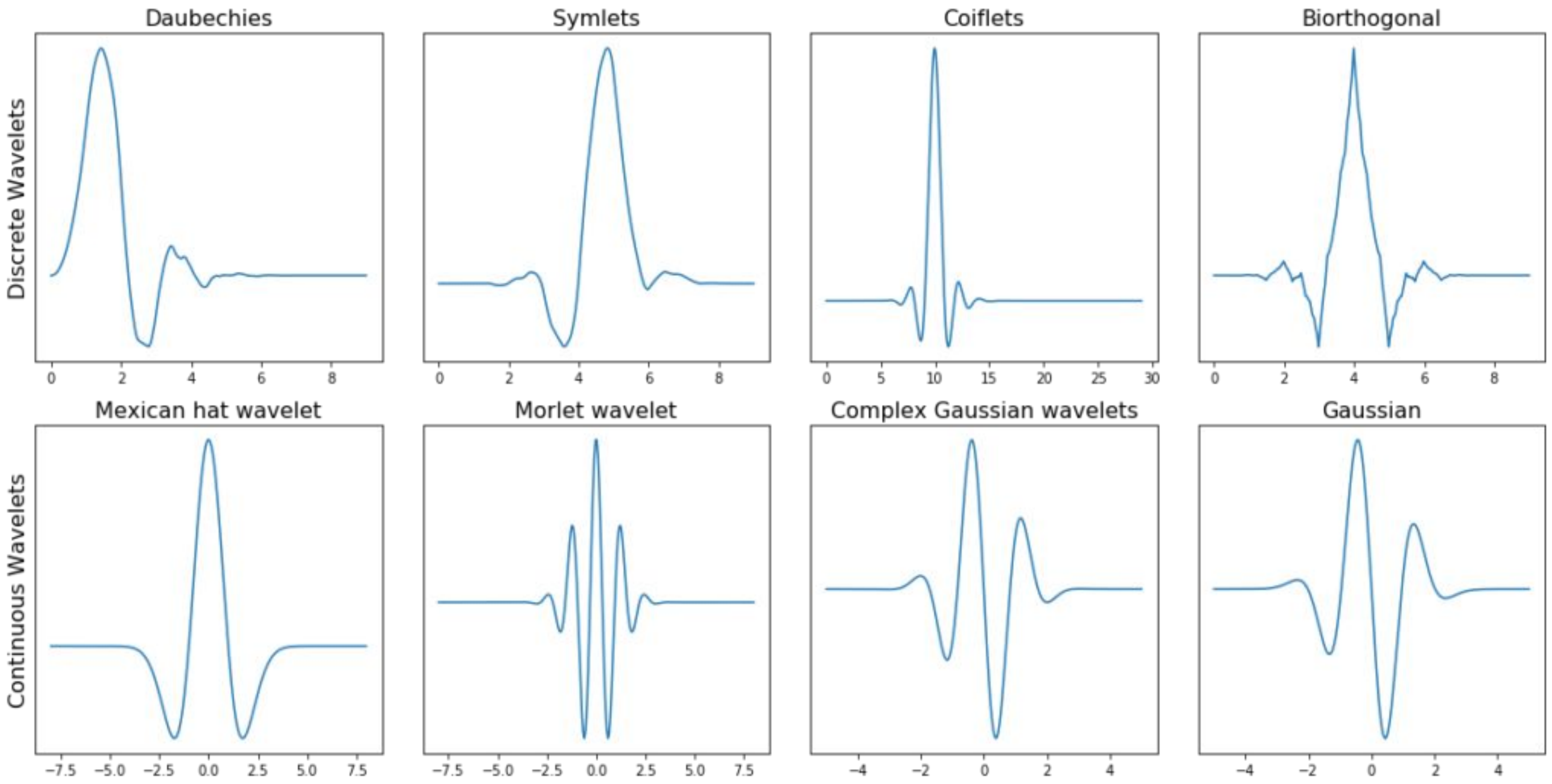
Our original series

The normalised parameter

mother wavelet

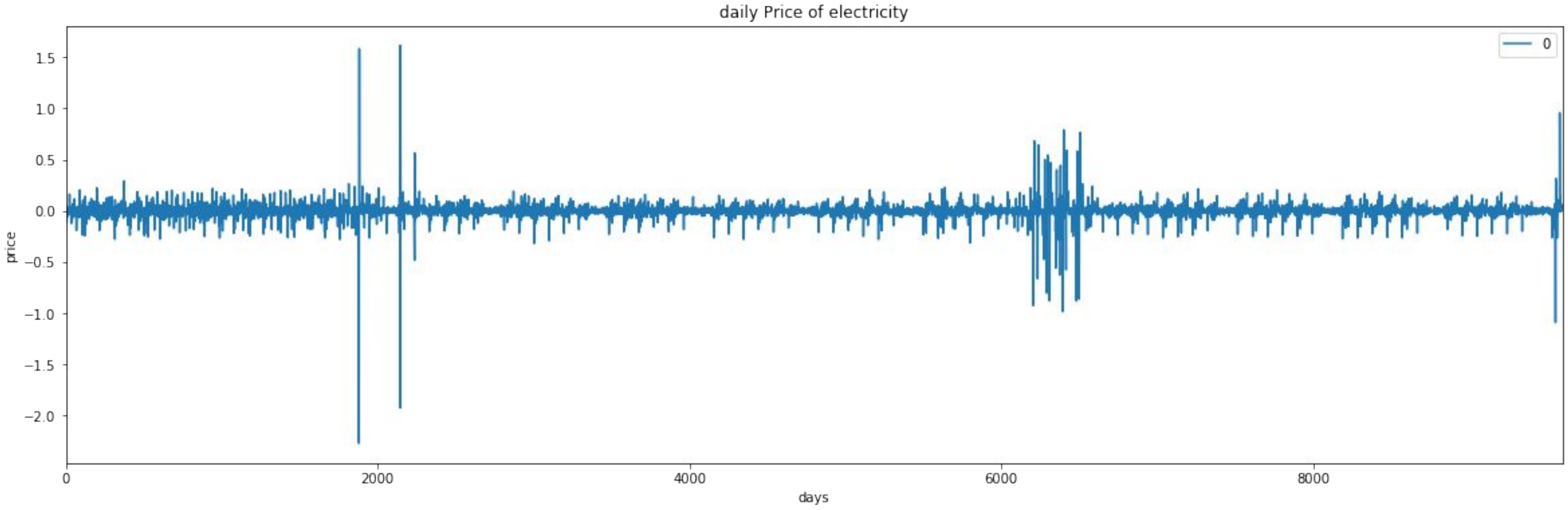
Sliding finite wave = mother wavelet(there are various types,used as per convenience
Window = Non zero magnitudes of the mother wavelet
Scale = size of the window

Different types of Wavelets..



Why we are not using it??

Sadly, we are not using this because our data is **already stationary**..
We need not decompose the series into more stationary components.



Extreme learning machine (ELM)

- Extreme learning machines are feedforward neural networks for classification, regression and clustering with a single layer or multiple layers of hidden nodes, where the parameters of hidden nodes need not be tuned.
- These hidden nodes can be randomly assigned and never updated.
- The output weights of hidden nodes are usually learned in a single step, which essentially amounts to learning a linear model.

How it works?

$(\mathbf{x}_i, \mathbf{y}_i)$ with M distinct samples, satisfied $\mathbf{x}_i \in \mathcal{R}^{d1}$ and $\mathbf{y}_i \in \mathcal{R}^{d2}$, the structure of ELM with N hidden neurons perfectly approximates to the given output as:

$$\sum_{i=1}^{N} \beta_i \mathbf{f}(\mathbf{w}_i^T \mathbf{x}_j + b_i) = \mathbf{y}_j, 1 \leq j \leq M \tag{1}$$

f: activation function; $\mathbf{w}_i$: the weight for connecting input layer and hidden layer; b_i: bias; β_i: output weight

How it works?

Eq. 1 can be written as $\mathbf{HB} = \mathbf{Y}$, $\mathbf{H}$ can be represented as:

$$\mathbf{H} = \begin{pmatrix} f(\mathbf{w}_1^T \mathbf{x}_1 + b_1) & \cdots & f(\mathbf{w}_N^T \mathbf{x}_1 + b_N) \\ \cdots & \cdots & \cdots \\ f(\mathbf{w}_1^T \mathbf{x}_M + b_1) & \cdots & f(\mathbf{w}_N^T \mathbf{x}_M + b_N) \end{pmatrix} \tag{2}$$

$\mathbf{B} = (\beta_1^T, \beta_2^T, ..., \beta_N^T)^T$ and $\mathbf{Y} = (y_1^T, y_2^T, ..., y_M^T)^T$.

B can be found by

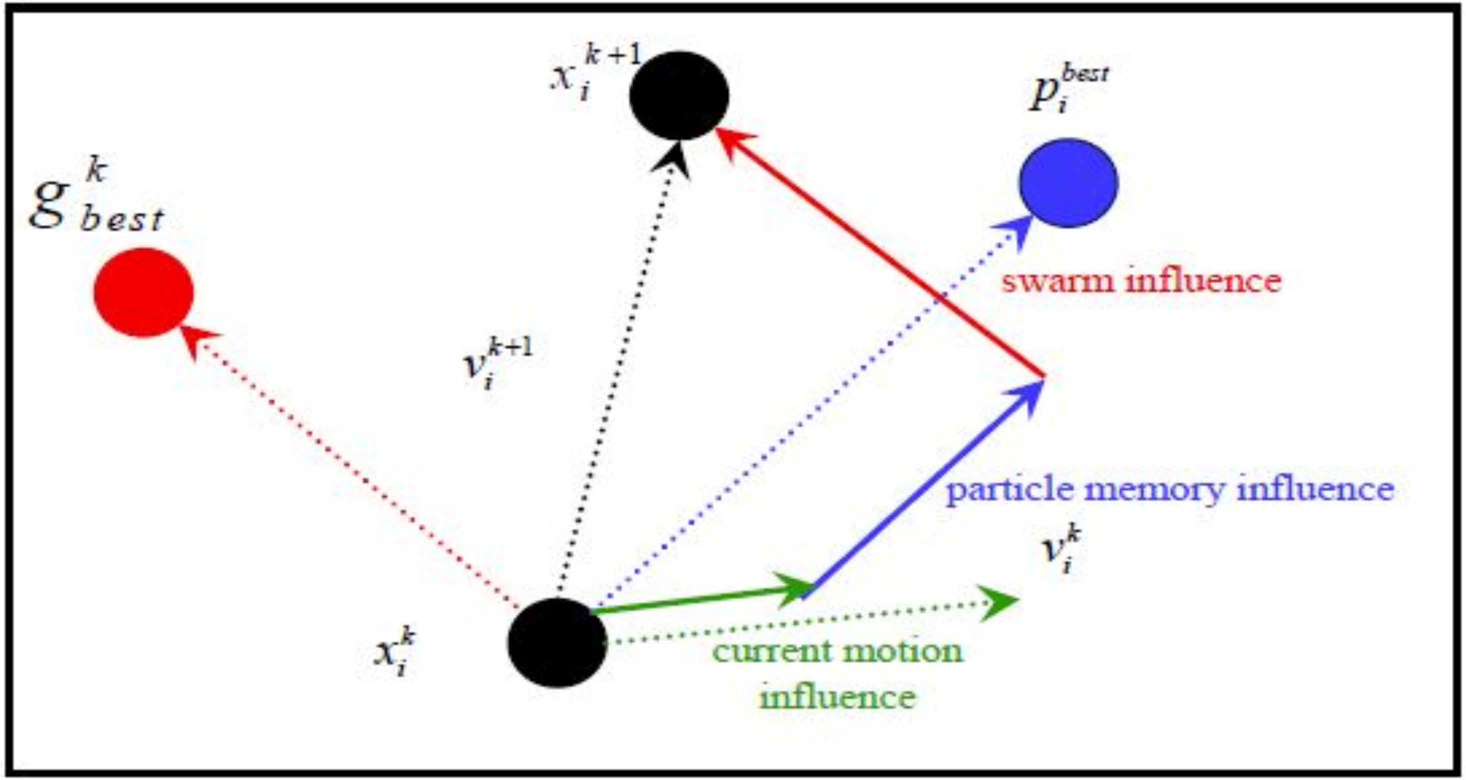
$$B = H^{-1} * Y$$

Particle Swarm Optimization (PSO)

- PSO was first intended for simulating social behaviour, as a stylized representation of the movement of organisms in a bird flock or fish school.
- PSO is a metaheuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions

How PSO works?

- Initially we have set of Candidate solutions (called Particles).
- These particles are moved around in the search-space
- Every Particle have three direction to move
  - Its current direction
  - Particle’s best direction
  - Swarm’s best direction



Visual representation of Single Particle movement

Mathematics of PSO

A particle i is defined by its position vector, x_i and its velocity vector, v_i. Every iteration, each particle changes its position according to the new velocity as

$$X_i(t+1) = X_i(t) + V_i(t+1)$$

Where

$$V_i(t+1) = w.V_i(t) + c_1 r_1 (xBest_i - X_i(t)) + c_2 r_2 (gBest - X_i(t))$$

xBest = best particle position
gBest = best swarm position
W = inertia weight

r1,r2 = random parameters in [0, 1]
c1,c2 = two positive constants

Time Series

Time Series is a set of observations on the values that a variable takes at different time.

Ex.Sales trend, Stock market prices, weather forecasts etc.

Stationarity

A series y_t is said to be stationary:

$E(X_t)=\boldsymbol{\mu}$

$Var(X_t)=\boldsymbol{\sigma}$^2

$Cov(X_{t,} X_{t+h})=f(h)$; does not depend on t

AR

Series current values depend on its own previous values
In equation form it can be written as

AR(1) : $y_t = a1 * y_{t-1}$

AR(2): $y_t = a1 * y_{t-1} + a2 * y_{t-2}$

AR(3): $y_t = a1 * y_{t-1} + a2 * y_{t-2} + a3 * y_{t-3}$

AR(p) - Current values depend on its own p-previous values

P is the order of AR process

MA

The current deviation from mean depends on previous deviations
MA(q) - The current deviation from mean depends on q- previous deviations
q is the order of MA process

MA(1): $\varepsilon_t = b1^* \varepsilon_{t-1}$

MA(2): $\varepsilon_t = b1^* \varepsilon_{t-1} + b2^* \varepsilon_{t-2}$

MA(3) :$\varepsilon_t = b1^* \varepsilon_{t-1} + b2^* \varepsilon_{t-2} + b3^* \varepsilon_{t-3}$

Differencing

Differencing in statistics is a transformation applied to time-series data in order to make it stationary.
A stationary time series properties do not depend on the time at which the series is observed.

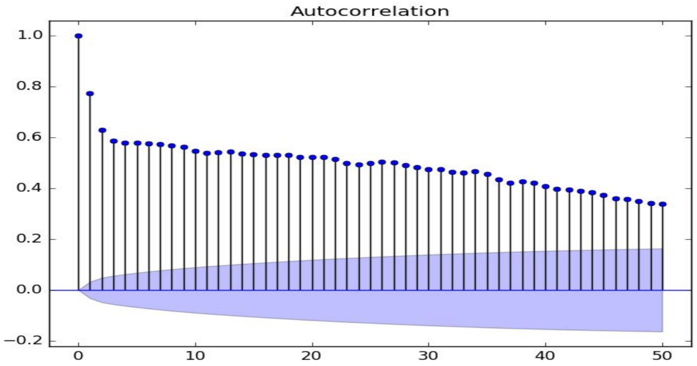
$y'_{t} = y_t - y_{t-1}$ One order differencing

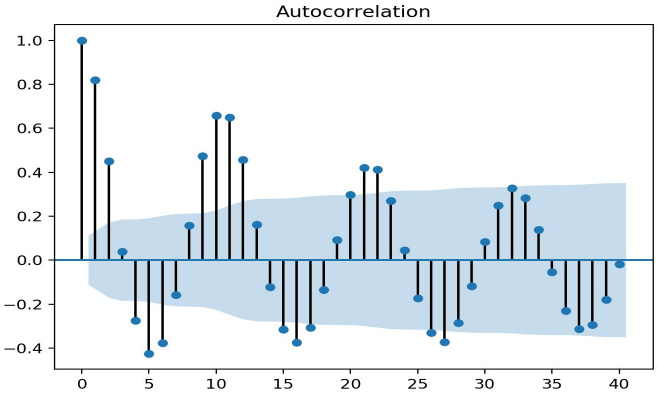
Autocorrelation

Correlation between series and its past values

Correlation between pairs of values at a certain lag

Use of Autocorrelation





ARIMA

Auto-**R**egressive **I**ntegrated **M**oving **A**verage

A ARIMA model is classified as an "ARIMA(p,d,q)" model, where:

p is the number of autoregressive terms,

d is the number of differences needed for stationarity, and

q is the number of lagged forecast errors in the prediction equation.

Performance Measure:

We have used three performance measure for checking how good our model is

1. Mean Absolute Error

$$MAE = \frac{1}{n}\sum_{i=1}^{n}\left|Y_i - y_i\right|$$

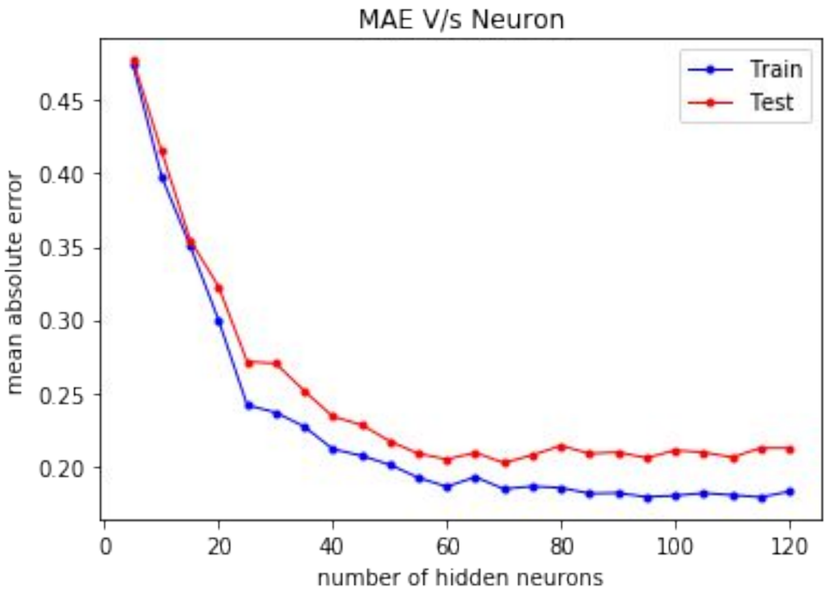
2. Mean Squared Error

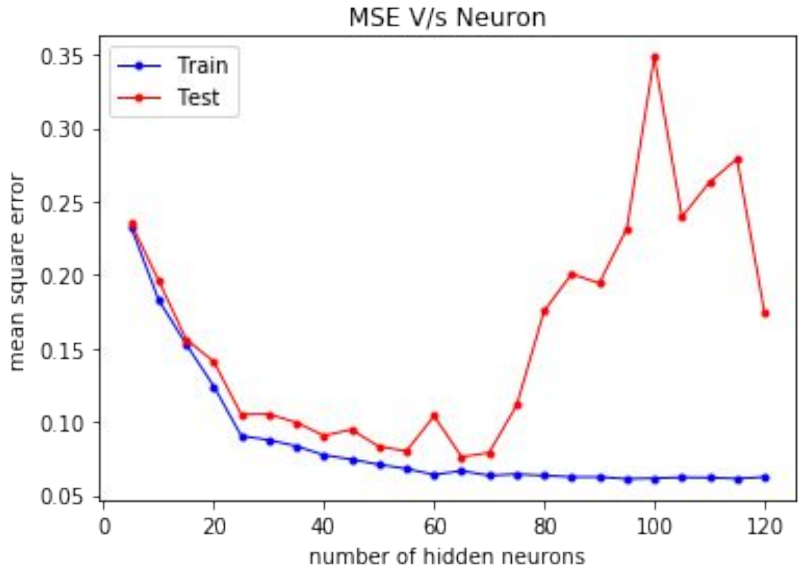
$$MSE = \frac{1}{n}\sum_{i=1}^{n}(Y_i - y_i)^2$$

3. R2 Score

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(Y_i - y_i)^2}{\sum_{i=1}^{n}(Y_i - y_{mean})}$$

Results of ELM





Results of various Methods :

Methods	***Mean Absolute Error***		***Mean Squared Error***		***R2 - Score***	
	Train	Test	Train	Test	Train	Test
EML	0.2012	0.2219	0.0717	0.0881	0.6014	0.6364
ARIMA	0.0350	0.0319	0.0032	0.0030	0.9810	0.9796
Hybrid (ELM+ARIMA)	0.0120	0.0112	0.0240	0.0225	0.8820	0.8745
ELM+PSO	-	-	0.0071	0.0130	-	-

Conclusion :

ARIMA gives the best forecast according to MSE and R2-Score

ELM+ARIMA gave best results on MAE.

While employing PSO optimiser on ELM,the model got overfit on train set and gave comparatively poor results on test set.

Reference :

1.Electricity price forecasting by a hybrid model, combining wavelet transform, ARMA and kernel-based extreme learning machine methods - ZhangYang[ab]LiCe[a]LiLian[a]
https://doi.org/10.1016/j.apenergy.2016.12.130

Thank You

QnA:

1.Why did we not use fourier transform?

A- The Fourier Transform has problem of resolution. You can either be sure of the frequency or the time of a signal, but not both.